

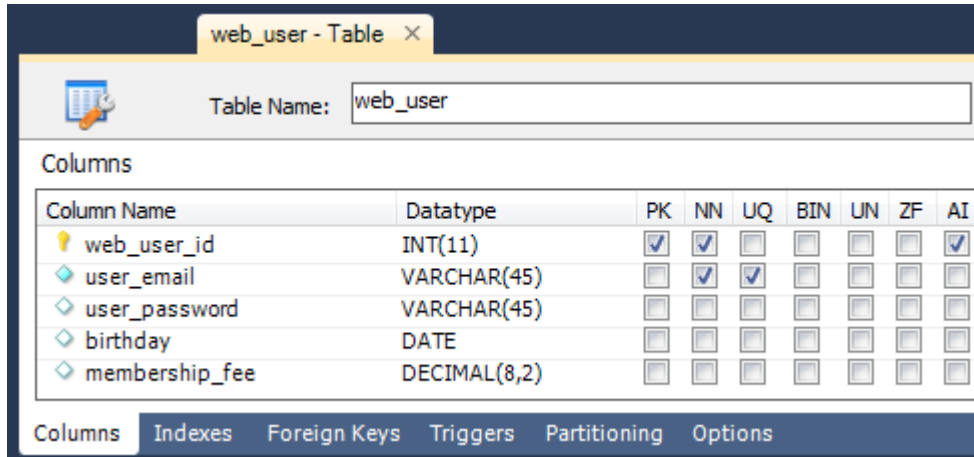
## Your Web Application Project Idea

In CIS 2308, each of you will be doing weekly labs that will culminate in your own individual web application. In order to make the labs create a simple but realistic web application, each student will have their own database, but all databases will have the same “shape”. This does limit the applications that each student can select, but there are a *lot* of options that fit the requirement. Each student database shall have a user table with a “many to many” relationship with another table. The many-to-many relationship is implemented by what we call an “associative table”. This database design allows for quite a bit of creativity, as you can see from the examples below. Come up with an idea for a database that will support your own web application. *Fill in the last row of this table with your idea.*

	"User table"	"Other table"	"Associative Table"		
What is the topic of the web app?	Who are the users that contribute content?	What “other table” do your users interact with?	What do users contribute to the database?	What can viewers of the site do?	Why is this a many-to-many relationship (between the "user table" and the "other table")?
Music Concerts	music band	concert hall	<b>Concert:</b> music band schedules a concert at a concert hall	search for concerts	A band can play at many different concert halls. A concert hall can host many different bands.
Wrestling Tournament Registration	wrestler	tournament	<b>Registration:</b> wrestler registers for a tournament	see who's registered for what tournaments	A wrestler can register for many tournaments. A tournament has many wrestlers registered.
Home Improvement Quotes	home owner	contractor	<b>Quote:</b> home owner requests quote from a contractor	see what quotes have been given to whom and for how much	A home owner can request quotes from several contractors. A contractor can provide quotes for several home owners.
course registration system	student	course	<b>Registration:</b> student registers for a course	generate class lists, create student schedule.	A student can take many courses. A course has many students enrolled.
Library Borrowing System	library patron	book	<b>Borrow:</b> library patron borrows a book	see what books have been borrowed and by whom	A library patron can borrow many books. A book can be (over time) borrowed by many library patrons.
Ecommerce	customer	product	<b>Purchase:</b> customer purchases a product	see who's bought which products.	A customer can purchase many products. A product can be purchased by many customers.
Sports Blog	sports fan	sports team	<b>Comment:</b> sports fan comments on a sports team	see other people's comments.	A fan can comment on many teams. A team can be commented on by many fans.
eCookbook	cook	food item (e.g., brownie)	<b>Recipe:</b> cook contributes a recipe for a food item	search for recipes	A cook can contribute recipes for many food items. A food item can have many different recipes.
Employee Benefit System	employee	benefit	<b>Selected Benefit:</b> employee selects a benefit	see who's signed up for which benefits	An employee can select many benefits. A benefit can be selected by many employees.
Travel Blog	traveler	country	<b>Trip:</b> traveler shares experiences about a trip to a country	search for travel experiences	A traveler can visit many countries. A country can be visited by many travelers.
Ecommerce	Vendor	Product	Purchase	See what products can be purchased	A vendor can purchase many products. A product can be purchased by many vendors.

## Define your database a little more.

1. **User Table.** Every student's database will have a table named exactly "**web\_user**" with the following layout. This is so that my sample code will work for you each week. In your web page, you may call your users other names (like customer or wrestler or traveler), but the data will be stored in the web\_user table.



The screenshot shows a database management interface window titled "web\_user - Table". It has a tabbed interface with "Columns" selected. The "Table Name" field contains "web\_user". Below the tabs, there is a table defining the columns for the "web\_user" table.

Column Name	Datatype	PK	NN	UQ	BIN	UN	ZF	AI
web_user_id	INT(11)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
user_email	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
user_password	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
birthday	DATE	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
membership_fee	DECIMAL(8,2)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

At the bottom, there are tabs for "Columns", "Indexes", "Foreign Keys", "Triggers", "Partitioning", and "Options".

Note: In a future lab, you can add more attributes into your web\_user table. You can also decide to not display some of the above attributes on your web pages (but you need to keep all the above attributes or sample code will not work for you.) The reason I am including birthday and membership\_fee is so that you can see code that deals with the date type and the decimal type (good for dollar amounts).

2. **Other Table.** Select a name for your "other table" and fill that in below (*it's like naming a variable, choose a short, representative name — you may not name it "other"*). Define the columns for your table following the requirements specified in the bullets below. Complete the diagram with your definitions (refer to the web\_user layout above as a sample).
  - An auto-increment Primary Key field. (*Use naming convention that PK column name is table name concatenated with "\_id", as was done for the web\_user table.*)
  - A unique, required, character field that describes or identifies the record. (*FYI: do not use column name "desc" because that is a sql keyword — short for "descending" — and it will cause problems when you code.*)
  - Add at least one non-character, optional field. (*Optional, meaning you do not click "not null" and this means that the user does not have to supply a value but can if they want.*)
  - You may add any other fields you like, but don't go crazy because it will result in too much work for you in future labs.

"Other" Table Name:	Product				
Column Name	Datatype	Primary Key?	Not Null?	Unique?	Auto-Increment?
Product ID	INT(11)	Yes	Yes	No	Yes
Product Name	VARCHAR(45)	No	Yes	Yes	No
Product Description	LONGTEXT	No	No	No	No
Unit Price	Decimal (3,2)	No	Yes	No	No

3. **Associative Table.** Select a name for your "associative table" (you may not call this table "associative", provide a representative name). Define the columns for your table following the requirements specified in the bullets below. Complete the diagram with your definitions.

- An auto-increment Primary Key field. (Use PK naming convention.)
- A required long character string (where people can add up to a few paragraphs of text).
- Add at least one non-character, optional field.
- Other fields (list field name, optional or required, data type, max chars if character). Add as many as you like, but don't go too crazy here either.

Associative Table Name:	Pricing Info				
Column Name	Datatype	Primary Key?	Not Null?	Unique?	Auto-Increment?
Pricing ID	INT(11)	Yes	Yes	No	Yes
Purchasing Info	LONGTEXT	No	No	No	No
Discount Rate	INT(3)	No	No	No	No
Phone Number	INT(10)	No	Yes	No	No

In a future lab, this table will get two add'l fields, a "pointer" to a particular web\_user and a "pointer" to a record from your other table.

4. Copy/Paste your web application idea from above, so you only have to print out this last page for me to check it. You can print in black and white if you want to.

	"User table"	"Other table"	"Associative Table"		
What is the topic of the web app?	Who are the users that contribute content?	What "other table" do your users interact with?	What do users contribute to the database?	What can viewers of the site do?	Why is this a many-to-many relationship (between the "user table" and the "other table")?
Ecommerce	Supplier	Product	Price	See what products can be purchased	A supplier can purchase many products. A product can be purchased by many suppliers.